



Banco de Dados para Jogos em Redes

Um banco de dados de videogame contém dados complexos em vários aspectos de cada jogo. Alguns bancos de dados começaram na década de 1990, tornaram-se populares e, de fato, continuam a prosperar até hoje. Quase sempre eram de origem coletiva e funcionavam como uma boa fonte para jogadores e entusiastas.

Com o passar do tempo, os bancos de dados de videogames continuaram a evoluir e deram origem a nichos de bancos de dados profissionais para atender a requisitos mais complexos. Eles são mantidos por equipes de especialistas em videogames, que cuidam muito da curadoria dos dados inseridos no banco de dados. Seu objetivo é fornecer dados de videogames de alta qualidade selecionados e classificados para o ecossistema de jogos.

> Os jogos digitais podem usar o banco de dados para armazenamento de dados e isso não significa que eles precisam necessariamente usar um SGBD para o gerenciamento, isso varia conforme a forma que os dados serão armazenados. De modo geral, para usar o SGBD deve ser usado conforme a complexidade da criação do game, e o armazenamento em massa e a forma que serão tratados os dados (OLIVEIRA e JUNIOR, 2022, p. 193).

A partir desse contexto, de acordo com RABIN (2012), podemos relacionar quais informações contém um banco de dados. Dentre os elementos,

podemos relacionar os seguintes itens:



- **Metadados de videogame (informações básicas)** – isso inclui informações como a data de lançamento do jogo, desenvolvedor/editor, classificações etárias, plataformas de lançamento e assim por diante. Os metadados do videogame são usados para identificar rapidamente as principais informações sobre um jogo, que podem ser usadas para encontrar jogos com dados semelhantes.

- **Divisão/desmontagem do jogo** – uma divisão/desmontagem de um jogo divide os vários conceitos e componentes que compõem um jogo, para analisar o que o faz funcionar. Isso se baseia nas informações incluídas nos metadados e adiciona dados como gênero, ações de jogo, conceitos de jogo (opções de design usadas no jogo), perspectiva, tipos de elementos usados (armas, mods e assim por diante). Com base nas informações coletadas em um detalhamento do jogo, os jogos podem ser classificados em grupos. Essa classificação ajuda a identificar padrões entre os jogos.

- **Multimídia relacionada ao jogo** – inclui capturas de tela, trailers de alta definição e vídeos de jogabilidade.

- **Relações** – essas informações analisam como um jogo se relaciona com outros de uma série ou franquia. Por exemplo, uma série inclui os projetos que continuam sua história com sequências. Um exemplo disso é a série Halo. Uma franquia é um grupo de jogos que giram em torno de um tema comum, mas possuem histórias individuais. Um exemplo disso é a franquia Assassin's Creed, que, curiosamente, também inclui uma série de sequências.

Essa mistura variada de informações enriquece o valor que um banco de dados pode fornecer, dando origem a muitas aplicações potenciais.

Linguagem SQL



De acordo com OLIVEIRA e JUNIOR (2022), um banco de dados é uma coleção de dados organizados para facilitar o gerenciamento. Ele é projetado para permitir a recuperação e o armazenamento, e o modo de pensar que define o arranjo dos dados é chamado de modelo de dados. Existem vários modelos de dados, e um dos principais é o modelo de dados relacional. Um banco de dados projetado de acordo com o modelo relacional é chamado de RDB (Relational Database - Banco de dados relacional).

Um banco de dados possui um “sistema de gerenciamento” para gerenciá-lo, e o sistema de gerenciamento do RDB é chamado de "sistema de gerenciamento de banco de dados relacional (RDBMS - Relational Database Management System).

Os bancos de dados são usados atualmente em vários campos e situações, como gerenciamento de informações de produtos e gerenciamento de informações de usuários de serviços.

A partir desse contexto, segundo OLIVEIRA e JUNIOR (2022), o SQL é uma abreviação de “Structured Query Language”. É uma linguagem de computador, mas não uma linguagem de programação. Uma linguagem para manipulação de dados em um RDB.

A linguagem que dá instruções ao banco de dados é chamada de instrução SQL, e o processamento é realizado pela combinação de instruções. Além disso, as instruções SQL são padronizadas por ANSI e ISO, portanto, podem ser usadas quase da mesma maneira com diversos Sistemas de Gerenciamento, por exemplo, Oracle Database, Microsoft SQL Server, My SQL, etc.

Segundo OLIVEIRA e JUNIOR (2022), no SQL você pode fazer coisas como:

- **Pesquisa condicional de dados:** você pode especificar condições detalhadas ao pesquisar. Isso é conveniente porque permite pesquisar dados com eficiência.

- **Aquisição/registro/atualização/exclusão de dados:** quando combinado com a pesquisa condicional de dados, você pode recuperar dados facilmente especificando uma linha ou coluna. Da mesma forma, registro, atualização e exclusão podem ser executados apenas para a linha especificada.

- **Criar/excluir/modificar uma tabela:** você pode criar, modificar e excluir tabelas conforme necessário, para poder configurar a tabela como desejar.

Como uma grande quantidade de dados é organizada pelo RDB, esses processos podem ser executados com eficiência. É necessário ter em mente a instrução SQL para o comando, mas depois de aprender o método, você poderá executar facilmente o processamento a partir daí.

Conexão com Banco de Dados: driver de conexão e ORM (Object-Relational Mapping)

Ao trabalhar com bancos de dados, muitas vezes surge a dúvida se uma pessoa deve usar linguagem de consulta estruturada (SQL) ou mapeamento objeto-relacional (ORM).

Segundo OLIVEIRA e JUNIOR (2022), o SQL permite que as pessoas interajam com bancos de dados relacionais usando consultas específicas. Um banco de dados relacional armazena informações em campos e geralmente organiza o conteúdo em linhas e colunas. Cada banco de dados geralmente contém uma ou mais tabelas.

O SQL é a linguagem de programação que permite ao indivíduo extrair ou modificar as informações desejadas ao trabalhar. Isso é feito por meio inserção de consultas que podem ser sensíveis ou não a maiúsculas e minúsculas. Por exemplo, o comando CREATE DATABASE cria um novo banco de dados, enquanto o comando INSERT INTO move as informações para um. Esses são dois tipos diretos de consultas. No entanto, dependendo das necessidades, eles podem ficar muito mais complicados. Nesses casos, os desenvolvedores podem estar mais suscetíveis a cometer erros se não estiverem atentos.

Como o ORM é diferente do SQL?

O mapeamento objeto-relacional permite que as pessoas interajam com bancos de dados em suas linguagens de programação preferidas, em vez de forçá-las a usar o SQL (FLYNN, 2022).

À medida que as interações das pessoas com os dados se tornam mais complexas, também aumenta o tamanho das consultas SQL a serem inseridas, o que pode aumentar a probabilidade de erros. Além disso, as coisas também ficam mais complicadas, pois existem várias maneiras de escrever perguntas e obter o mesmo resultado. Um mapeador objeto-relacional funciona como um tradutor que converte o código de uma forma para outra. A ferramenta cria objetos que representam um mapa virtual do banco de dados. O usuário pode interagir com os objetos em vez de se envolver diretamente com o código (FLYNN, 2022).

Integração de Banco de Dados com a Unity3D

A Unity é uma game engine que facilita o processo de criação de jogos digitais e aplicativos móveis, construídos sobre as ferramentas de desenvolvimento nativas de várias plataformas como Windows, Android e iOS, entre outras. Isso significa que você pode criar e enviar aplicativos pertencentes a essas plataformas. O propósito existencial da Unity é desenvolver jogos que exijam mais renderização gráfica do que

processamento de dados. A Unity suporta oficialmente a linguagem de programação C#.



Segundo Pardalis (2020), a Unity é uma plataforma de desenvolvimento de jogos versátil e rica em recursos. Embora o suporte a um banco de dados incorporado não seja trivial no Unity, a plataforma permite que você faça isso de maneira simples por meio do sistema de plug-ins.

Capturar e transmitir dados em redes não confiáveis é uma tarefa difícil. É inútil capturar dados se você não conseguir entregá-los de forma segura e consistente ao destino escolhido. Uma forma de garantir a entrega segura e consistente dos dados é ter algum tipo de mecanismo de cache para armazenar os eventos. Esse mecanismo armazena os eventos até que você receba a confirmação de que os dados foram entregues corretamente ao destino.

É possível armazenar eventos no Unity usando PlayerPrefs. No entanto, podemos relacionar algumas restrições que essa prática gera, por exemplo:

- Devemos ser capazes de armazenar um buffer de eventos de tamanho arbitrário.

- Devemos ter flexibilidade de definir e impor um esquema nos eventos que são armazenados. PlayerPrefs é um armazenamento de valor-chave simples que suporta um conjunto muito limitado de tipos de dados e nenhuma hierarquia.

- Gerenciar o buffer não deve acontecer no thread principal. Trabalhar com PlayerPrefs acontece no thread principal, o que não é uma opção viável na maioria dos casos.

Em vez de PlayerPrefs, a melhor opção é utilizar o SQLite para essa finalidade. O SQLite é um mecanismo de banco de dados SQL leve,

incorporável e altamente confiável que funciona incrivelmente bem para dispositivos móveis.



Os bancos de dados são, essencialmente, estruturas de dados que funcionam sob certo conjunto de regras que nós mesmos definimos. Eles nos libertam da agitação do microgerenciamento, mas dão a liberdade de moldar os dados de acordo com nossas necessidades. Existem dois tipos famosos de bancos de dados: um é chamado de Linguagem de Consulta Estruturada ou SQL, que organiza os dados na forma de tabelas. O outro é chamado de No-SQL, porque não possui tabelas ou estrutura, ele armazena dados na forma de objetos. Intuitivamente, o No-SQL consome mais recursos do que o SQL, portanto, para aplicativos móveis, normalmente as empresas preferem o SQL.

Falando em recursos, bancos de dados foram feitos para computadores e servidores para armazenar grandes quantidades de dados. No entanto, no caso de plataformas móveis, não temos realmente esse tipo de recursos de computação e claramente também não precisamos de tanta funcionalidade. É aqui que entra o SQLite, oferecendo menos funcionalidade para economizar recursos computacionais. O SQLite é leve, possui menos tipos de dados e não permite funcionalidades como acesso multiusuário.

O SQLite é um dos Sistemas de Gerenciamento de Banco de Dados preferidos pelos desenvolvedores móveis e recomendado pela documentação oficial do Android.

Atividade Extra

A partir do que foi apresentado neste módulo, principalmente no processo de integração do banco de dados em um jogo digital, o trabalho acadêm intitulado "**BANCO DE DADOS NA PERSPECTIVA DO DESENVOLVIMENTO DE JOGOS**

DIGITAIS" (OLIVEIRA E SANTOS Jr., 2022), disponível no Google Acadêmico, apresenta que os jogos digitais necessitam da utilização de estratégias de armazenamento de dados, banco de dados, SGBDs, com o objetivo de salvar e guardar dados do usuário e também do estado do jogo, dados, etc. Ao longo do projeto será possível refletir sobre o desenvolvimento de um jogo, visando demonstrar uma das possibilidades que podem ser adotadas na integração entre banco de dados e jogos digitais.

Referência Bibliográfica

ARRUDA, E. P. **Fundamentos para o Desenvolvimento de Jogos Digitais**. Porto Alegre: Grupo A, 2014.

FLYNN, S. **What Is the Difference Between SQL and Object-Relational Mapping (ORM)?**. 2022. Disponível em: < https://www.kdnuggets.com/2022/02/difference-sql-object-relational-mapping-orm.html#:~:text=Someone%20who%20does%20not%20use,shy%20away%20from%20learning%20SQL >. **(Acesso em: 17/05/2023)**

OLIVEIRA, P.; JUNIOR, A. **Banco de dados na perspectiva do desenvolvimento de jogos digitais**. Rio de Janeiro, 2022. Disponível em: < https://www.researchgate.net/profile/Antonio-Junior-60/publication/360398444_BANCO_DE_DADOS_NA_PERSPECTIVA_DO_DESENVOLVIMENTO_DE_JOGOS_DIGITAIS/links/62865f948ecbaa07fcc19880/BANCO-

DE-DADOS-NA-PERSPECTIVA-DO-DESENVOLVIMENTO-DE-JOGOS-DIGITAIS.pdf >. **(Acesso em: 17/05/2023)**



PARDALIS, K. **Using SQLite on Unity for Android and iOS**. 2020. Disponível em: <https://www.rudderstack.com/blog/mobile-persistent-storage-with-sqlite-on-unity-for-android-and-ios/ >. **(Acesso em: 17/05/2023)**

RABIN, S. **Introdução ao Desenvolvimento de Games**. São Paulo: Cengage Learning Brasil, 2012.

Ir para exercício